ENHANCING MARITIME DOMAIN AWARENESS: A COMPARATIVE STUDY OF HYBRID TRANSFORMER ARCHITECTURES FOR ROBUST VESSEL AIS TRAJECTORY PREDICTION

Alexander Schiøtz *Felix Thomsen* *Bertram Hage* *Christian Rand*
s221221 *s221710* *s224918* *s224930*

Technical University of Denmark

ABSTRACT

This paper addresses the challenge of forecasting vessel trajectories using Automatic Identification System (AIS) data from the Danish maritime authority (Søfartsstyrelsen). We introduce a MapReduce pipeline and K-means sampling strategy to mitigate "open-sea bias" and capture diverse maneuvers. Validating a kinematic Kalman Filter against a discrete TrAISformer and our custom continuous hybrid CNN-Transformer (TPTrans), we identify a critical trade-off: while TPTrans achieves superior short-term precision (lowest 1-hour ADE), TrAISformer demonstrates greater long-term stability and navigational realism. Ultimately, the navigational realism inherent in discrete generative architectures like TrAISformer establishes the necessary foundation for trustworthy, autonomous maritime surveillance. Code: `https://github.com/alexander9908/AIS-MDA`.

1. INTRODUCTION

The Automatic Identification System (AIS) provides a vast amount of spatio-temporal data essential for monitoring global shipping. However, raw AIS streams are characterized by irregular sampling rates, noise, and transmission gaps, making trajectory prediction a complex sequence modeling task. Accurate forecasting is critical for collision avoidance and anomaly detection in crowded maritime corridors like the Danish straits. Vessel trajectory prediction is an ongoing field of study, with data-driven deep learning methodologies playing an increasingly prominent role [1, 2]. Recent advancements in spatio-temporal data modeling have shown that attention mechanisms and Transformer-based architectures offer superior performance in handling long-term dependencies and complex traffic interactions compared to traditional Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) variants [3, 4, 5]. In this work, we present a comprehensive deep learning framework for trajectory prediction. Our contributions include a scalable preprocessing pipeline for large-scale AIS logs and a robust K-means window sampling strategy that ensures diverse training coverage based on traffic density clusters. Furthermore, we conduct a comparative analysis of three modeling paradigms: a kinematic Kalman Filter, a discrete generative Transformer (TrAISformer), and a continuous regression Hybrid Transformer (TPTrans).

2. DATA AND PREPROCESSING

We utilize historical AIS data provided by the Danish Maritime Authority (Søfartsstyrelsen) [6]. The dataset covers vessel movements observed via AIS stations deployed on Danish shores and data is available since 2006. In this project we use 3 months of data in the period July 24th - October 24th. The raw data includes 1,818,441,680 AIS messages from 35,349 unique vessels based on their unique Maritime Mobile Service Identity (MMSI) number. The data includes basic geographical and kinetic features such as latitude, longitude, Speed over Ground (SOG) in knots, and Corse over Ground (COG) in degrees. Additionally, the data includes further demographic features such as vessel type and navigational status, although these features lack data completeness especially for smaller vessels. We include just latitude, longitude, SOG and COG as features for our modeling.

2.1. Data preprocessing

We wish to turn the irregular sampled and noisy single messages into regularly sampled valid sequences. We focus or study to model larger vessels, namely tankers and cargo ships, and thus filter out messages not marked as either type. We furthermore exclude messages outside our defined geographical boundary of $lat \in [54, 59]$, $lon \in [5, 17]$ and filter out messages not conforming to our boundary of $SOG \in [0, 30]$ and $COG \in [0, 360]$, resulting in 445,395,128 remaining messages. We split the data by MMSI and perform further processing in parallel [7], reducing computation time and space, while avoiding lossy simplification such as the otherwise popular Ramer–Douglas–Peucker algorithm [8]. Following [3], we define a voyage as a sequence of messages with time gaps smaller than 2 hours. We partition voyages longer than 20 hours and discard those shorter than 4 hours or containing fewer than 20 messages. Additionally, we remove outliers where the calculated speed exceeds 40 knots. Finally, to ensure regular sampling, we apply linear interpolation at 5-minute intervals and min-max normalize all features. These steps reduced the dataset from 72,995 to 41,458 voyages spanning 1,203 unique vessels.

3. METHODOLOGY

We frame the vessel trajectory prediction task as a multivariate sequence-to-sequence regression problem. Given an observed sequence of historical states $X = \{x_1, \ldots, x_T\}$ where $x_t \in \mathbb{R}^4$, the goal is to predict the future sequence $Y = \{y_{T+1}, \ldots, y_{T+L}\}$ based on the feature set $F = \{\text{Latitude, Longitude, SOG, COG}\}$. During training we fix $T = 64$ (past), $L = 12$ (future). As a non-learning baseline, we implement a standard Kalman Filter assuming a Constant Velocity (CV) model based on the vessel's last known heading and speed [9, 10].

3.1. Data Sampling Strategy: K-Means Clustering

Since many trajectories in the dataset exceed the required sequence length $T + L$, we extract training samples from the full vessel paths with an oversampling of 4. However, because commercial vessels often follow established shipping lanes, standard uniform sampling would disproportionately favor dense and low speed segments. To ensure the model is exposed to a diverse set of geographical segments, we implement a cluster-based sampling strategy. We fit a K-Means algorithm ($k = 50$) on the coordinates of all preprocessed AIS messages and for a given trajectory uniformly sample a cluster and starting point within that cluster (Appendix Section 7.2).

3.2. TrAISformer (Generative Discrete)

We adapt the TrAISformer architecture proposed by D.Nguyen et al. in [3], a decoder-only Transformer architecture utilizing multi-head self-attention to capture long-term temporal dependencies. To address the multimodal nature of vessel trajectories, the authors discretize the four continuous features (latitude, longitude, SOG, and COG) into respective $N_{feature}$ bins, and construct a sparse "four-hot" vector h_t by concatenating the one-hot vectors of the four features. The sparse vector h_t is subsequently projected into a dense, high-dimensional continuous embedding vector e_t. During

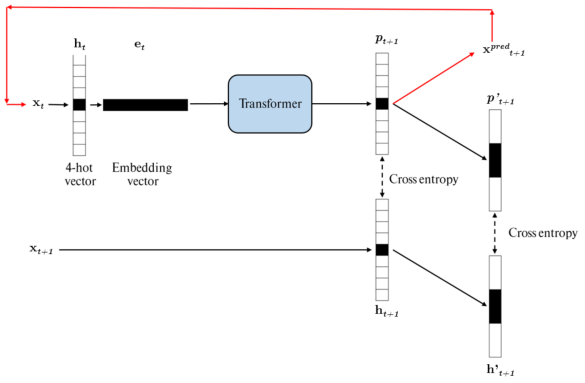


Fig. 1: Sketch of the TrAISformer architecture. Source: [3]

training, the Transformer's output is split into four components corresponding to each feature. The total loss is derived from the sum of their individual cross-entropies plus the same cross-entropy from a coarse-resolution version (merging 3 consecutive bins when constructing h_t), which is weighted by a β hyperparameter:

$$\mathcal{L}_{CE} = \sum_{l=1}^{L} CE\left(p_{T+l}, \mathbf{h}_{T+l}\right) + \beta CE\left(p'_{T+l}, \mathbf{h}'_{T+l}\right), \quad (1)$$

where $p_{T+l} = p(h_{T+l}|e_{0:T+l-1})$, h'_{T+l} is the coarse target with $\dim(h'_t) = \lfloor \frac{\dim(h_t)}{3} \rfloor$, and p'_{T+l} is the coarse prediction derived by applying a fixed $[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ smoothing convolution.

During inference, a h^{pred}_{t+1} is sampled from the Transformers output distribution p_t with temperature T, and a y_{t+1} is reconstructed from h^{pred}_{t+1} by using the center of the bins. The predicted y_{t+1} is then fed back into the network to predict the following prediction step until y_{T+L} is obtained.

For our implementation we set $N_{lat} = N_{lon} = 128$ and $N_{SOG} = N_{COG} = 64$. We use 4 attention heads, 4 decoder layers, an embedding dimension of 512 and a dropout of 0.1 during training. Initial tests with larger model architectures yielded no improvement in generalization.

3.3. TPTrans (Continuous Hybrid Regression)

Our primary contribution is a custom implementation of the TPTrans model [4], which performs continuous regression of coordinate displacements via three stages:

1. Local-Global Encoder: The input sequence X is processed by a 1D Convolutional block ($Conv1d \rightarrow ReLU \rightarrow Conv1d$) to extract local kinematic derivatives (velocity, acceleration) and smooth sensor noise. The result is augmented with sinusoidal positional encodings and passed to an N-layer Transformer Encoder, which uses multi-head self-attention to capture long-range dependencies across the 64-step history.

2. Query-Based Decoder: Unlike recurrent architectures, we utilize a Transformer Decoder with L learnable query embeddings $Q \in \mathbb{R}^{L \times d_{model}}$. These queries interact with the encoder's memory via multi-head *cross-attention*, allowing the model to attend to specific historical maneuvers (e.g., turn entries) to predict future steps $t + 1 \ldots t + L$ in parallel.

3. Scaled Delta Learning: To ensure stability, inputs are MinMax normalized to $[0, 1]$, and the model regresses inter-step displacements $\Delta\hat{y}_t$. Predictions are denormalized during the forward pass and optimized using Huber Loss against scaled ground-truth targets to prevent gradient underflow:

$$\mathcal{L}_{TPTrans} = \text{Huber}(\Delta\hat{y}_{pred}, \Delta y_{true} \cdot \alpha) \quad (2)$$

where $\alpha = 100.0$ is a scaling factor applied to physical degree differences.

Ablation: GRU vs. Transformer Decoder and Model Specification

We also implemented a TPTrans variant using a Gated Recurrent Unit (GRU) decoder (see Appendix Section 4). However, it underperformed compared to the Transformer decoder, which we attribute to the Information Bottleneck problem: the GRU is forced to compress the entire 64-step history into

a single fixed-size context vector [11, 12]. Consequently, the Transformer Decoder was selected for the final TPTrans architecture (Fig. 2). Furthermore, we evaluated different model capacities during training. We found that a medium-sized architecture offered the optimal trade-off between performance and generalization. For the final TPTrans implementation, we set the latent dimension $d_{model} = 512$ with 8 attention heads, 8 encoder layers, and 8 decoder layers.

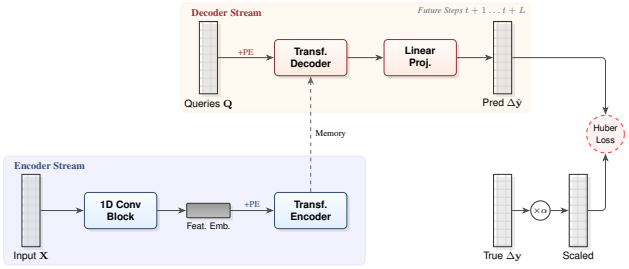


Fig. 2: Architecture of the TPTrans model. The dual-stream design separates the historical context encoding (bottom blue stream) from the future trajectory decoding (top orange stream). The decoder utilizes learnable queries combined with cross-attention to the encoder's memory to regress displacement deltas in parallel. (PE = Positional Encoding)

3.4. Inference and Evaluation Strategy

Constrained Decoding via Water Guidance
A fundamental challenge in maritime trajectory prediction is ensuring navigational validity; purely kinematic models often predict trajectories that cross landmasses in narrow straits. To enforce physical constraints, we implement a post-hoc **Water Guidance** mechanism. We utilize a high-resolution rasterized water mask of the Danish Exclusive Economic Zone (EEZ). During inference, for every predicted location $\hat{y}_{t+1}$, we verify if the vector connecting $\hat{y}_t$ to $\hat{y}_{t+1}$ intersects land pixels. If a violation is detected, $\hat{y}_{t+1}$ is projected to the nearest valid water coordinate using a Euclidean distance transform. This ensures that all evaluated models benefit from topographical awareness, providing a fair comparison of kinematic performance.

Deterministic Evaluation & Rolling Horizon
While [3] employs a "Best-of-N" sampling strategy ($N = 16$), selecting the best prediction post-hoc, we reject this as unsuitable for real-world maritime surveillance (e.g., Real-time trajectory prediction, port prediction, anomaly detection and collision avoidance), where the true future is unknown at inference time. Instead, we enforce strict **deterministic evaluation** to assess reliability: **TrAISformer** uses greedy decoding ($T = 0$) to select the most probable bin; **TPTrans** regresses the deterministic mean of displacement deltas; and the **Kalman Filter** projects linearly without noise. To handle long horizons, we employ autoregressive generation: TrAISformer predicts step-by-step ($t \to t+1$), while TPTrans operates *block-autoregressively*, forecasting full 1-hour horizons ($H = 12$) iteratively by appending predictions to the history window.

Evaluation Metrics
We evaluate performance in kilometers (km) using the Haversine distance function $d_{hav}(\cdot)$. We report the *Average Displacement Error (ADE)*, measuring the mean path deviation, and the *Final Displacement Error (FDE)*, assessing destination accuracy at step $T + L$:

$$ADE = \frac{1}{L}\sum_{t=1}^{L} d_{hav}(\hat{y}_{T+t}, y_{T+t}), FDE = d_{hav}(\hat{y}_{T+L}, y_{T+L}) \tag{3}$$

4. RESULTS & DISCUSSION

4.1. Experimental setup

The dataset was partitioned into training, validation, and test sets in an 80:10:10 ratio split by MMSI to avoid data leakage and prevent the model from overfitting to individual vessels' movement patterns. We train using the AdamW optimizer with a batch size of 512 and an initial learning rate of 3×10^{-4} on a single NVIDIA A100 (80GB) GPU [13]. Training runs for a maximum of 200 epochs with early stopping (patience 50), utilizing a 5-epoch warm-up followed by validation-based learning rate decay (factor 0.5) triggered by loss stagnation.

4.2. Quantitative Analysis

Table 1 summarizes the predictive performance. We observe a distinct trade-off between short-term precision and long-term robustness. **Short-Horizon Precision (1 Hour):** The **TPTrans** model achieves the lowest Mean ADE (1.21 km) and FDE (2.07 km), demonstrating that its continuous regression head combined with the CNN encoder effectively smooths sensor noise and captures immediate kinematic trends. Interestingly, the **Kalman Filter** achieves the lowest *Median* ADE (0.66 km). This indicates that for the majority of "easy" cases (linear, steady-state navigation), simple kinematics remains highly effective. However, its significantly higher Mean error reveals its catastrophic failure on non-linear maneuvers. **Long-Horizon Robustness (3 Hours):** As the prediction horizon extends, the autoregressive drift in the continuous TPTrans model accumulates, leading to higher errors (Mean $FDE_{3h} = 11.69$ km). In contrast, the generative **TrAISformer** demonstrates superior long-term stability, achieving the lowest ADE and FDE at 3 hours (Median $ADE_{3h} = 2.24$ km). We attribute this to its discrete classification objective; by predicting probability distributions over a fixed grid, the model is less prone to "drifting off into open space" and tends to keep predictions "snapped" to learned high-density shipping lanes, even if exact timing precision is lost.

4.3. Qualitative Discussion

Maneuver Handling: In complex scenarios, such as sharp turns within narrow passages (Fig. 3a, 3f), the Kalman filter fails catastrophically, projecting a straight line directly into land due to its lack of environmental awareness. **TPTrans** generally initiates maneuvers correctly with smooth curvature, but predictive accuracy tends to degrade during sharp

Table 1: Quantitative trajectory prediction results in kilometers (km). **Bold** indicates the best performance in that category. Note the trade-off between TPTrans' short-term precision and TrAISformer's long-term stability.

Model	1 Hour Horizon		2 Hour Horizon		3 Hour Horizon	
	ADE (Mean/Med)	FDE (Mean/Med)	ADE (Mean/Med)	FDE (Mean/Med)	ADE (Mean/Med)	FDE (Mean/Med)
Kalman Filter	1.58 / **0.66**	3.60 / 1.50	3.70 / 1.77	8.86 / 4.64	6.05 / 3.38	14.84 / 9.21
TrAISformer	1.38 / 0.90	2.41 / 1.37	2.62 / **1.48**	**5.42 / 2.70**	**4.20 / 2.24**	**9.41 / 4.53**
TPTrans (Ours)	**1.21** / 0.79	**2.07 / 1.23**	**2.55** / 1.50	5.85 / 2.92	4.61 / 2.60	11.69 / 6.28
Dataset Statistics: Mean Trip Length = 4.17 h, Median Trip Length = 4.42 h						

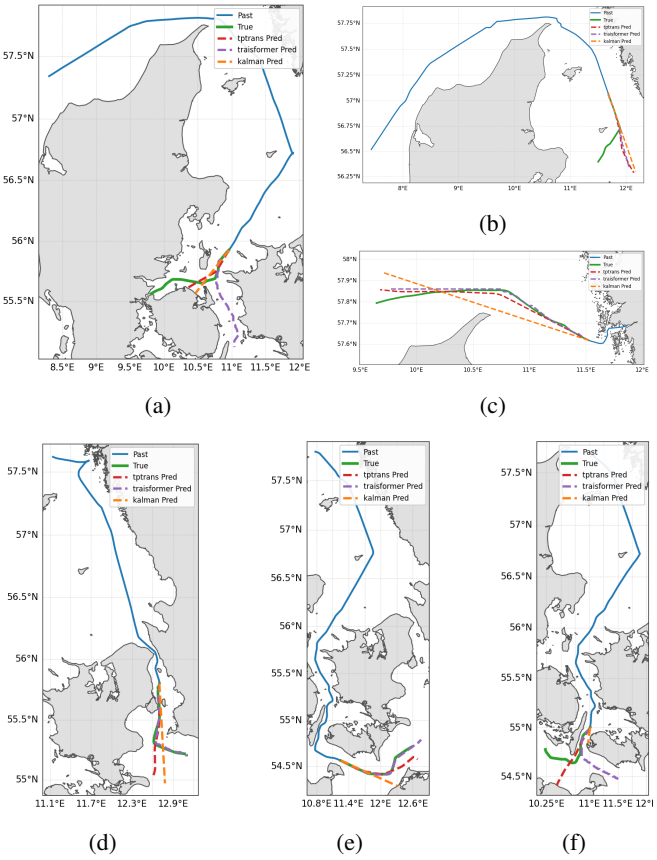


Fig. 3: Qualitative results (Trip cut 80%). **Blue**: Past, **Green**: True (1h). **Red**: TPTrans, **Purple**: TrAISformer, **Orange**: Kalman.

turns later in the forecast horizon (Fig. 3c–3f). Conversely, **TrAISformer** captures turning dynamics with high precision (Fig. 3c–3e. However, it highlights the challenge of long-term intent prediction: it occasionally "hallucinates" a plausible but incorrect alternative route, as seen in Fig. 3a and 3f, where it predicts a valid shipping lane that the vessel did not actually take. In cases of highly anomalous behavior that deviates from historical traffic patterns (Fig. 3b), all models struggle. They tend to regress towards the statistically most probable path rather than the specific, irregular maneuver executed by the vessel.

Realism vs. Precision: The evaluated architectures exhibit distinct trade-offs. The Kalman Filter excels strictly in linear, open-water segments (Fig. 3c) where kinematic inertia dominates. **TPTrans** minimizes Euclidean displacement error, offering robust initial predictions and excelling in open waters with gradual curvature, though it struggles to execute sharp turns deep into the prediction horizon. In contrast, **TrAISformer** predictions are characterized by "blocky" zig-zag artifacts due to spatial discretization. Its performance is often binary: it either captures the trajectory perfectly or diverges completely onto an alternative *valid* lane. These results fundamentally challenge standard evaluation paradigms: should prediction quality be defined by Euclidean proximity to the ground truth, or by the navigational plausibility of the forecast? While TPTrans offers superior numerical precision, TrAISformer demonstrates a critical advantage in generating semantically valid trajectories that respect maritime boundaries and shipping lanes (Fig. 3a,3b,3f). For real-world maritime surveillance and collision avoidance, where predicting a physically possible maneuver is paramount, TrAISformer's ability to model the most probable, realistic path arguably renders it the superior architecture despite its discretization artifacts.

5. CONCLUSION

In this work, we validated a hybrid CNN-Transformer (TPTrans) and a discrete generative model (TrAISformer) against a kinematic baseline for AIS trajectory prediction. While the Kalman filter suffices for linear segments, deep learning architectures trained with density-based K-means sampling proved essential for modeling complex non-linear maneuvers. Crucially, our findings challenge standard evaluation paradigms in maritime forecasting. Quantitatively, TPTrans achieves the highest precision (lowest Mean ADE) at the 1-hour horizon, yet TrAISformer demonstrates superior stability over longer durations. We argue that Euclidean error minimization does not fully capture predictive utility; despite discretization artifacts, TrAISformer exhibits superior *navigational realism*. It consistently predicts valid trajectories constrained to shipping lanes, whereas regression models often minimize error by averaging into non-navigable space. Consequently, TrAISformer's ability to model the multimodal distribution of valid vessel routes renders it the more robust architecture for maritime surveillance. Future work will aim to bridge this gap by incorporating environmental data and exploring Graph Neural Networks for vessel interactions.

ACKNOWLEDGEMENTS

We would like to thank our supervisor, Peder Heiselberg from the Technical University of Denmark (DTU), for his valuable guidance, motivation, and assistance.

6. REFERENCES

[1] Xiaocai Zhang, Xiuju Fu, Zhe Xiao, Haiyan Xu, and Zheng Qin, "Vessel trajectory prediction in maritime transportation: Current approaches and beyond," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, 2022.

[2] Jinqiang Bi, Hongen Cheng, Wenjia Zhang, Kexin Bao, and Peiren Wang, "Artificial intelligence in ship trajectory prediction," *Journal of Marine Science and Engineering*, vol. 12, no. 5, 2024.

[3] Duong Nguyen and Ronan Fablet, "Traisformer: A transformer network with sparse augmented data representation," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[4] Wentao Wang, Wei Xiong, et al., "Tptrans: Vessel trajectory prediction model based on transformer using ais data," *ISPRS International Journal of Geo-Information*, 2024.

[5] Nicos Evmides, Michalis P. Michaelides, and Herodotos Herodotou, "Vessel trajectory prediction with deep learning: Temporal modeling and operational implications," *Journal of Marine Science and Engineering*, vol. 13, no. 8, 2025.

[6] Danish Maritime Authority (DMA), "Ais data," `https://www.dma.dk/safety-at-sea/navigational-information/ais-data`, 2025, Accessed: November 21, 2025.

[7] Jeffrey Dean and Sanjay Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, 2008.

[8] Urs Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer graphics and image processing*, vol. 1, no. 3, 1972.

[9] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, 1960.

[10] Assaf Mansour, Voicu Groza, and Emil Petriu, "The use of kalman filter techniques for ship track estimation," *WSEAS TRANSACTIONS ON SYSTEMS*, 2020.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2015.

[12] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[13] DTU Computing Center, "Dtu computing center resources," `https://doi.org/10.48714/DTU.HPC.0001`, 2025.

[14] OpenAI, "Chatgpt," Large language model.

[15] GitHub, "Github copilot," AI coding assistant.

DECLARATION OF USE OF GENERATIVE AI

During the development of this project, we utilized generative AI (GAI) tools to support specific research and implementation tasks. **ChatGPT** [14] assisted in the initial literature search and concept clarification; all resulting information and cited references were manually verified by the authors for accuracy. GAI tools were employed exclusively for refining sentence structure and improving readability. No full sentences or paragraphs were generated by AI; all text was critically reviewed, modified, and finalized by the authors. Finally, **GitHub Copilot** [15] provided code suggestions and documentation retrieval to expedite development. AI-generated code constitutes a minor portion of the final codebase and was rigorously audited for correctness and functionality.

AUTHOR CONTRIBUTIONS

All authors (Alexander Schiøtz (**s221221**), Felix Thomsen (**s221710**), Bertram Hage (**s224918**), and Christian Rand (**s224930**)) contributed equally to the conceptualization, methodology, software implementation, validation, and writing of this report.

Specific technical contributions were distributed as follows: Christian Rand and Bertram Hage spearheaded the data engineering, K-means sampling strategy, and the MapReduce preprocessing pipeline. Alexander Schiøtz led the implementation and training of the deep learning architectures, with support from Bertram Hage and Felix Thomsen. Alexander Schiøtz and Christian Rand designed the deterministic evaluation framework and conducted the quantitative analysis. Bertram Hage and Felix Thomsen developed visualization utilities and provided ad hoc technical support and code optimization throughout the project lifecycle.

7. APPENDIX

7.1. Alternative TPTrans GRU Architecture

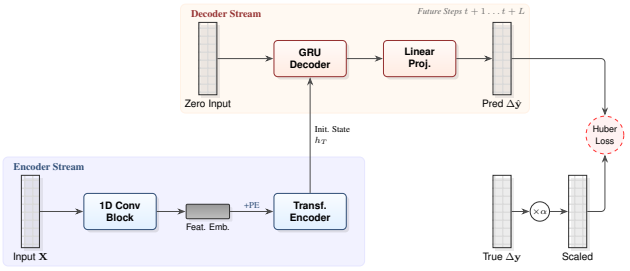


Fig. 4: Architecture of the TPTrans-GRU variant. The Encoder remains identical to the Transformer model, but the Decoder is replaced by a recurrent GRU. Instead of cross-attention, the GRU is initialized with the final hidden state h_T of the Encoder and unrolls the future trajectory autoregressively from a zero-input start.

7.2. K-Means Sampling Visualization

This section provides visualizations of our K-Means sampling strategy, which is crucial for mitigating the "open-sea bias" inherent in AIS data (Fig. 5, 6 and 7).

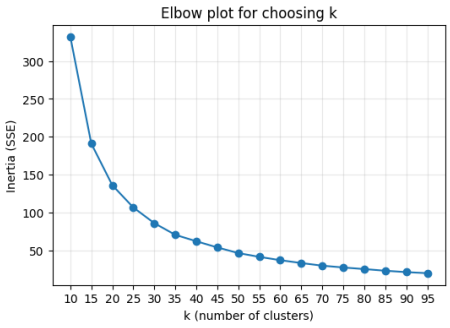
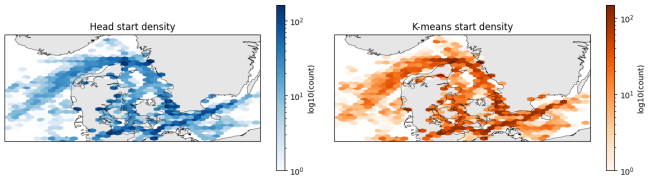
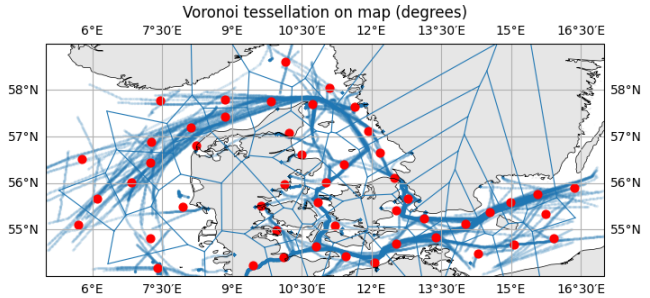


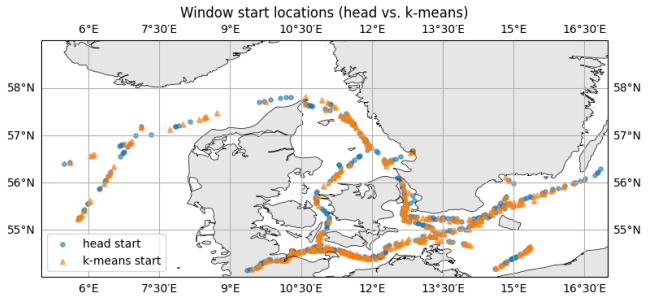
Fig. 5: Hyperparameter selection for K-Means clustering using the Elbow Method. We analyze the inertia (sum of squared distances to the nearest centroid) as a function of k. The curve exhibits an "elbow" behavior where marginal gains diminish significantly after $k = 30$. We selected $k = 50$ as a conservative trade-off to ensure sufficient granularity for capturing diverse maneuvering patterns in complex straits without over-fragmenting the spatial domain.



(a) Comparison of start-point density. **Left:** Standard head-start sampling heavily favors ports (blue). **Right:** Our K-Means sampling strategy (orange) successfully redistributes focus to high-density clusters near ports and turning points. (Example data: 5 days)



(b) Voronoi tessellation of the coordinate space based on $k = 50$ centroids (red dots). The blue lines represent the boundaries between clusters, ensuring that training samples are drawn from diverse geographical regions. (Example data: 5 days)

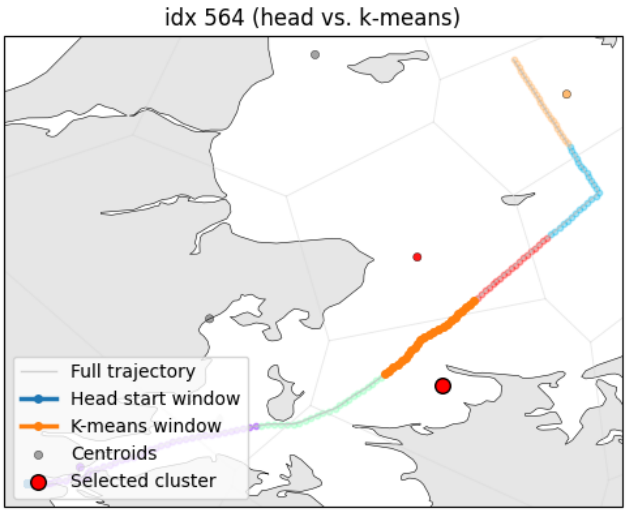


(c) Spatial comparison of training window start locations. Standard **Head-Start** sampling (Blue circles) disproportionately favors voyage origins. In contrast, the proposed **K-Means** strategy (Orange triangles) successfully redistributes training samples across the spatial domain, ensuring coverage of critical mid-voyage maneuvers and high-density straits. (Example data: 5 days)

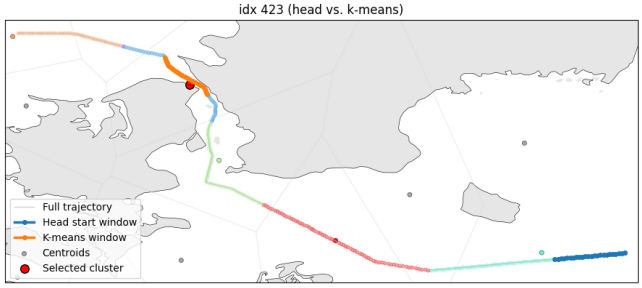
Fig. 6: Analysis of the K-Means clustering strategy used to balance the training distribution.

7.3. Additional Qualitative Results (trajectory prediction plots)

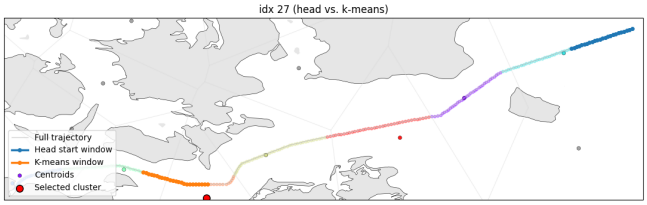
We provide additional examples of model predictions on the test set in Fig. 8 to further illustrate the performance differences discussed in Section 4.



(a) Example trajectory (ID 564). The orange segment highlights a window selected via K-Means sampling.

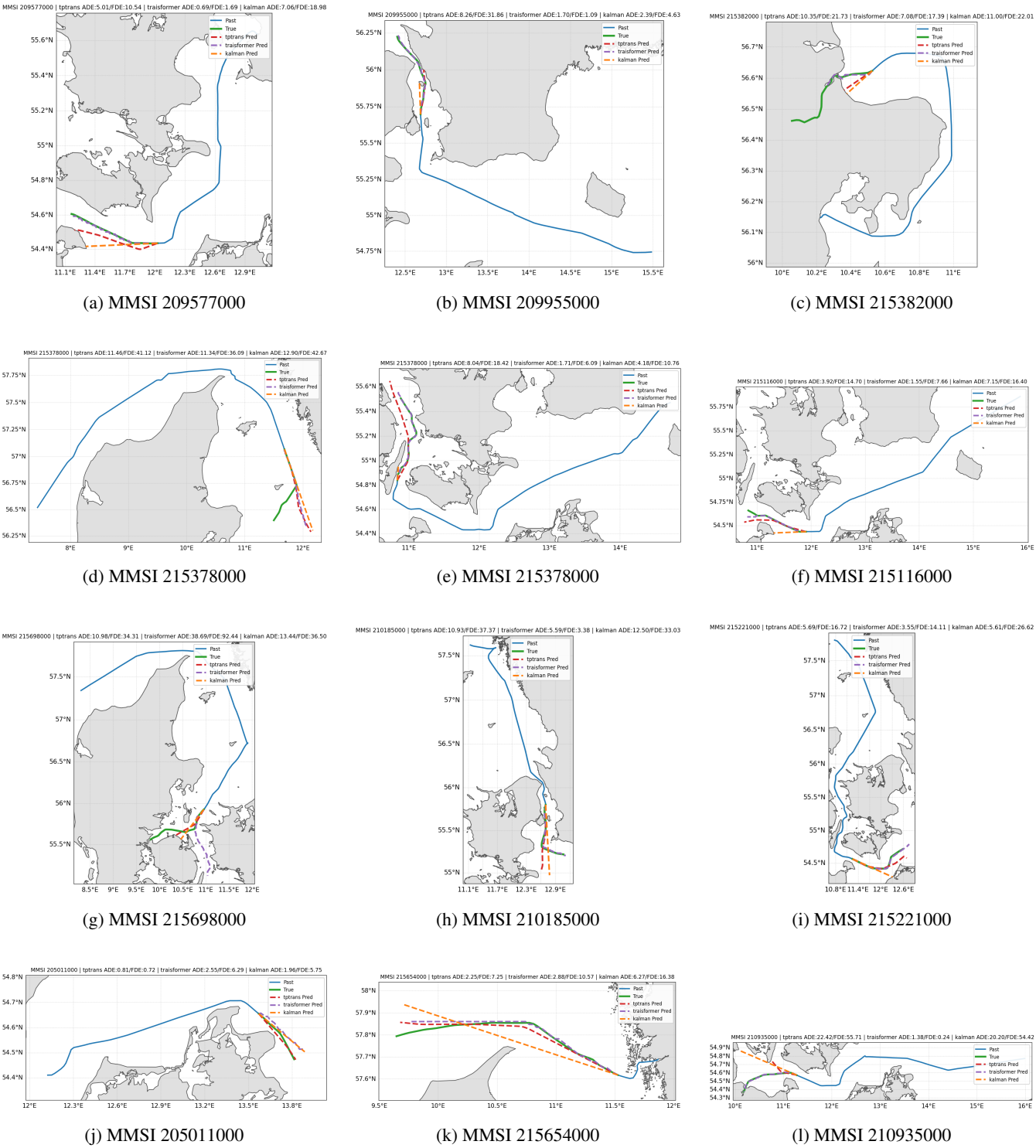


(b) Example trajectory (ID 423). The sampling strategy prioritizes the complex maneuver near the coastline (orange segment).



(c) Example trajectory (ID 27). K-Means sampling ensures coverage of the strait passage.

Fig. 7: Qualitative examples of training windows selected by the K-Means strategy (Orange) versus standard Head-Start sampling (Blue). The selected cluster centroid is marked in red.



(a) MMSI 209577000 (b) MMSI 209955000 (c) MMSI 215382000

(d) MMSI 215378000 (e) MMSI 215378000 (f) MMSI 215116000

(g) MMSI 215698000 (h) MMSI 210185000 (i) MMSI 215221000

(j) MMSI 205011000 (k) MMSI 215654000 (l) MMSI 210935000

Fig. 8: Extended qualitative results comparing TPTrans, TrAISformer, and Kalman Filter predictions on various test trajectories.